

## DNA Computing – A Future Facet of Bioinformatics?

Tom Royce

DNA computing attracts much attention both because of its peculiarity and the possibility of solving NP (and other difficult) problems in a timely fashion. DNA computing offers data storage capacity, energy efficiency, and parallelism unthinkable with today's conventional computational devices. DNA computing is generally not accepted as a topic of bioinformatics. This may be true for the current nascent state of the field but as milestones are reached I believe DNA computing will play a strong role in bioinformatics both because the techniques can solve difficult combinatorial problems and for the future possibility of building 'wet' DNA sequence databases. For the field to advance, the diverse skill set possessed by bioinformaticians is desperately needed. The aim of this paper is to introduce the field of DNA computing.

Dr. Leonard Adleman first demonstrated DNA's computing capabilities in 1994 by solving a simple case of the Hamiltonian Path Problem. Recall that this problem takes as input a directed graph  $G$ , a start vertex  $v$  and destination vertex  $v'$ . The question to be answered is, "Does  $G$  contain a path  $P$  from  $v$  to  $v'$  such that  $P$  visits every node in the graph exactly once." Adleman solved this problem for a graph containing seven nodes as follows.

First, consider the following operations:

**AMPLIFY** – sequences  $p_1$  and  $p_2$  – Make many copies of DNA molecules that contain the two subsequences  $p_1$  and  $p_2$ . Only the region of the molecules inclusively between  $p_1$  and  $p_2$  are copied. This is accomplished via PCR.

**EXTRACT** – sequence  $p$  – extract from solution only those molecules that contain subsequence  $p$ . This can be done with sequence-specific streptavidin-coated beads.

**SELECT** – length  $i$  – select only those molecules of length  $i$

Assign each node  $n$  in the graph a different 20-mer sequence  $S(n)$ . Put these sequences in a test tube (there are actually millions of copies of each sequence). Also, for each edge from some node  $m$  to some node  $n$ , create 20-mers such that its initial 10-mer is complementary to the terminal 10-mer of  $m$  and its terminal 10-mer is complementary to the initial 10-mer of  $n$ . (Again, there are millions of these 'edge' molecules). Allowing these molecules to bind with each other creates all possible paths through the graph in parallel.

Next, amplify all molecules beginning with  $v$  and ending in  $v'$ . So, only molecules starting at the start node and ending at the destination node are amplified. From these molecules, **SELECT** those that have length 140. Since a 20-mer sequence represents each node, a molecule of length 140 corresponds to a path that visits seven nodes (the number of nodes in Adleman's graph) exactly once. From these molecules, **EXTRACT** those that contain the sequence corresponding to some node in the graph. From these

extracted molecules, EXTRACT those that contain a sequence corresponding to another, different node in  $G$ . Repeat this EXTRACT process for each node in  $G$ . The molecules that are extracted in the final EXTRACT step must therefore contain Hamiltonian paths by definition.

Adleman's work demonstrated the feasibility of computing with DNA. Richard Lipton built upon Adleman's techniques to solve an instance of the Satisfiability Problem (SAT) soon after. Since then, SAT has traditionally been the problem of choice for new molecular computation techniques. Recently, a twenty variable 3-SAT problem has been solved which included twenty-four clauses. Despite the success of such a large-scale computation, there has been concern that the computations require much laborious effort from bench scientists. For DNA computation to be useful, autonomous methods are needed.

Masami Hagiya's lab first proposed a state transition machine implemented with DNA. (Their technique is often referred to as the 'whiplash model.')

Since a state transition machine can model Turing machines, these machines can solve any computational problem, theoretically.

The state transition machine consists of a single-stranded DNA molecule of the form  $5'-X(s_1')(s_1)X(s_2')(s_2) \dots X(s_n')(s_n)(\text{spacer})-3'$  where  $X$  represents a sequence that stops DNA polymerase from copying, spacer is a long sequence of random DNA,  $s_n$  represents the state before a transition and  $s_n'$  represents the state following the transition. The initial state,  $C(s_n)$  for some state  $s_n$ , is appended onto the 3' end of the machine. The sequence  $C(s_n)$  then binds to its corresponding state in the molecule, forming a hairpin structure. DNA polymerase then uses  $C(s_n)$  as a primer and appends  $C(s_n')$  onto the 3' end of  $C(s_n)$ . This process theoretically continues until a state is encountered that has no associated transition.

Ehud Shapiro and co-workers through the implementation of automata achieved further progress in autonomous DNA computing. The input string is represented as DNA strings as done in previous experiments. For example if the 6-mer sequence CTGGCT represents an 'a' and CGCAGC represents a 'b' then an input sequence could look like:

$$I = \text{CTGGCTCTGGCTCGCAGCCTGGCT.}$$

Also, a termination symbol is encoded with DNA at the end of the sequence. Essentially, each input symbol is recognized by a 'transition molecule' that (along with an enzyme) cleaves off that symbol and sets up the next symbol to be read by other transition molecules. Eventually, all of the input string is consumed leaving the termination symbol exposed. A special 'output detector' molecule recognizes which state the automata finishes in. Hence, the user can identify whether or not the machine terminated in an accept state.

Hopefully, the three experiments sketched here have introduced the current state of DNA computing. It is unlikely that these systems will replace silicon-based computers but

their study is important in that they represent an alternative viewpoint to conventional computing. Besides an interesting quirk in the history of computer science, DNA computing may have some interesting applications such data encryption and portable 'wet databases' of enormous scale. It is also foreseeable that DNA computation could play a role in the development of bacterial cell lines with novel and programmable computational power. Such technology could have a huge impact on the field of bioinformatics and on the biotechnology industry.